Utilizando Inteligência Artificial como tutor para estudantes de programação no intuito de auxiliar na aprendizagem de novas tecnologias

Arthur J. Oliveira[1]

[1]Pontifícia Universidade Católica de Minas Gerais

arthur.jansen@sga.pucminas.br

Abstract. *The growth of the technology sector has increased the demand for professionals specialized in specific programming languages. This study proposes the use of LLMs (Large Language Models) as a didactic support tool for students transitioning between programming stacks. The research was conducted with volunteers who completed three coding tasks and answered a questionnaire to assess the impact of AI on learning. With this, fill gaps in the literature and contribute to the discussion on the use of AI (Artificial Intelligence) in technical education.*

Resumo. *O crescimento da área de tecnologia tem ampliado a demanda por profissionais especializados em linguagens específicas. Diante disso, este trabalho propõe o uso de LLMs (Large Language Models) como ferramenta de apoio didático para estudantes em transição entre stacks de programação. A pesquisa foi realizada com voluntários que desenvolveram três códigos e responderam a um questionário para avaliar os impactos do uso da IA (Inteligência Artificial) na aprendizagem. Com isso, preencher lacunas na literatura e contribuir para a discussão sobre o uso de IA na educação técnica.*

Bacharelado em Engenharia de *Software* - PUC Minas
Trabalho de Conclusão de Curso (TCC)

Orientador de conteúdo (TCC I): João Leatisteli - joao.leatisteli@sga.pucminas.br
Orientador de conteúdo (TCC I): Joana Souza - joanasouza@pucminas.br
Orientador de conteúdo (TCC I): Leonardo Vilela - leonardocardoso@pucminas.br
Orientador acadêmico (TCC I): Cleiton Tavares - cleitontavares@pucminas.br
Orientador do TCC II: Ramon Lacerda Marques - ramonmarques@pucminas.br

Belo Horizonte, 23 de Novembro de 2025.

1. Introdução

Nos últimos anos, a Inteligência Artificial tem se difundido amplamente no Brasil, alcançando um público que, em grande parte, ainda está em processo de letramento digital sobre seu funcionamento. Muitos usuários iniciantes de ferramentas de geração de texto, como GPT e Gemini, desconhecem os princípios que sustentam essas tecnologias. Esse desconhecimento pode levar ao uso inadequado e à subutilização de suas capacidades [Brasil 2025]. Dado que muitos usuários empregam essas ferramentas em contextos de

aprendizagem, é fundamental a adoção de boas práticas que orientem um uso apropriado e consistente com as finalidades educacionais previstas.

Nesse cenário, os *Large Language Model* (LLMs) têm se destacado como ferramentas capazes de apoiar o aprendizado por meio de respostas rápidas e personalizadas [Frankford et al. 2024]. Entretanto, sua confiabilidade ainda é limitada, pois LLMs podem gerar informações imprecisas, fenômeno conhecido como 'alucinação' [Bang et al. 2023, Li et al. 2023]. Portanto, diante desses desafios associados ao uso de LLMs, este trabalho tem como problema central investigar **se a utilização de uma LLM pode efetivamente favorecer o aprendizado de uma nova *stack* (conjunto de tecnologias)**.

Com os avanços recentes em Inteligência Artificial, especialmente com as *LLMs*, torna-se viável o uso dessas ferramentas no apoio ao ensino de programação. Esses modelos oferecem respostas, análise e correção de código, beneficiando iniciantes e estudantes em transição entre linguagens. Um estudo aponta impactos positivos: 78% dos alunos relataram auxílio significativo do tutor em IA (Inteligência Artificial), com aumento nas taxas de aprovação nas matérias cursadas por eles [Ma et al. 2024]. Em cursos de engenharia de software, seu uso é promissor, desde que acompanhado por supervisão humana [Frankford et al. 2024]. Contudo, ainda é necessário reduzir riscos de alucinação [Phung et al. 2024] para garantir que essas ferramentas cumpram seu papel de auxiliar na construção de fundamentos, o que é um elemento essencial no aprendizado de novas tecnologias [Shrestha and Parnin 2019]. Embora existam estudos sobre o uso de tutores em IA no ensino de programação, poucas pesquisas avaliam empiricamente a eficácia de LLMs na aprendizagem de novas *stack*.

Frente a esse cenário, este estudo tem como objetivo geral **avaliar o impacto do uso de uma LLM no processo de aprendizagem de novas *stacks* de programação**. Para isso foram definidos os seguintes objetivos específicos: i) desenvolver um sistema capaz de auxiliar na aprendizagem de uma nova *stack* de programação; ii) aplicar o sistema em um experimento controlado envolvendo desafios práticos de programação; iii) analisar, de forma quantitativa e qualitativa, os resultados obtidos no experimento, avaliando a eficácia do sistema no processo de aprendizagem.

Como principal contribuição, este trabalho busca avaliar os impactos positivos e negativos associados ao uso do Sistema Tutor, observando a qualidade e utilidade dos *feedbacks* gerados aos estudantes de programação. Espera-se que a ferramenta gere *feedbacks* úteis, coerentes e tecnicamente adequados, contribuindo para a compreensão conceitual e prática dos estudantes durante os desafios propostos. Além disso, o estudo pretende estabelecer uma base consistente para pesquisas futuras e aplicações práticas, auxiliando programadores e educadores a tomarem decisões mais conscientes sobre a integração de IA em processos de estudo e formação.

Por fim, o artigo é organizado da seguinte forma: a Seção 2 aborda a Fundamentação Teórica; a Seção 3 apresenta os trabalhos relacionados; a Seção 4 evidencia os materiais e métodos; a Seção 5 Resultados; a Seção 6 Discussão dos Resultados e Ameaça a validade; a Seção 7 Conclusão.

2. Fundamentação Teórica

Nesta seção, apresentam-se os conceitos fundamentais para compreender o trabalho.

2.1. Alucinações em LLMs

O uso de modelos de linguagem como o GPT-2 apresenta avanços em tarefas de linguagem natural, mas também traz desafios, entre eles o fenômeno conhecido como alucinações. Esse termo descreve situações em que o modelo gera respostas que, embora pareçam coerentes, estão incorretas ou são inventadas. Frequentemente, os modelos produzem respostas com informações erradas, especialmente em tarefas factuais, como responder perguntas ou resumir textos [Radford et al. 2019]. Essas limitações revelam que, apesar do bom desempenho geral, os modelos ainda carecem de mecanismos robustos para validar a veracidade de suas saídas.

Radford (2019), autores do estudo, reconhecem esse problema ao analisar o desempenho do GPT-2 em tarefas como leitura e compreensão de textos, geração de resumos e respostas factuais. Em um dos testes, por exemplo, o modelo afirma que "Califórnia"é o maior estado dos EUA em área, quando a resposta correta seria o Alasca, demonstrando um erro factual mesmo com aparente confiança na resposta. Esses exemplos evidenciam que, embora os LLMs apresentem respostas de forma fluida, não possuem compreensão semântica real nem acesso a verificação de fatos, o que torna o uso consciente e crítico dessas ferramentas essencial para evitar interpretações equivocadas.

2.2. Modelos de Linguagem de Grande Porte

Modelos de linguagem de larga escala são sistemas baseados em inteligência artificial que apresentam capacidades avançadas de compreensão e geração textual. Embora tenham sido inicialmente desenvolvidos para tarefas voltadas ao processamento de linguagem natural, os LLMs evoluíram a ponto de assumirem funções relevantes no contexto do desenvolvimento de software, como na geração automática de código, no preenchimento de lacunas e na detecção de erros em programas. De acordo com Wang e Chen (2023), esses modelos são treinados em conjuntos de dados, o que lhes permite interpretar instruções em linguagem natural e convertê-las em trechos de código.

Essa capacidade de produzir código contextualizado resulta da estrutura sofisticada dos LLMs e do grande número de parâmetros que os compõem. No entanto, como ressaltam Smolié (2024), a verificação da qualidade e precisão dos códigos gerados ainda exige acompanhamento humano, especialmente para mitigar riscos relacionados à segurança e à funcionalidade das aplicações. Assim, embora os LLMs representem um avanço significativo na automação de atividades de programação, ainda enfrentam desafios quanto à confiabilidade do código gerado e à sua adequação às particularidades de diferentes linguagens e padrões de desenvolvimento [Jiang et al. 2024].

2.3. *Prompt Engineering*

Prompt engineering é uma técnica fundamental no uso de modelos de linguagem como o ChatGPT, pois permite que o usuário formule instruções claras e estratégicas em linguagem natural para obter respostas precisas e úteis. Essa prática vai além de comandos simples, ela envolve estruturar os *prompts* com intenção, contexto e formato adequados, guiando o modelo a responder conforme objetivos específicos. Na área de desenvolvimento de software, por exemplo, essa abordagem tem sido usada para automatizar tarefas, gerar códigos ou revisar sistemas com eficiência e precisão.

De acordo com White (2024), o *prompt engineering* se estabelece como forma de programação em linguagem natural e representa uma habilidade essencial para desenvolvedores que buscam explorar todo o potencial dos modelos de linguagem. Os autores destacam que o uso de padrões estruturados chamados *prompt* patterns contribui na interação com LLMs, padronizando a entrada e garantindo saídas confiáveis e relevantes para tarefas computacionais específicas.

3. Trabalhos Relacionados

Nesta seção, são apresentados artigos relacionados a este trabalho com o objetivo de aprofundar o conhecimento na área de aprendizagem de estudantes de Engenharia de Software com o uso de Inteligência Artificial. Além de destacar técnicas voltadas à melhoria e limpeza de *prompts*, serão considerados estudos com resultados tanto conclusivos quanto inconclusivos, especialmente aqueles que enfrentaram dificuldades na execução e na validação dos dados para fins educacionais.

Os autores de *Ma et al.* (2024) propõem a criação de uma tecnologia baseada em modelos como GPT, *DeepSeek* e Gemini os quais são LLMs que computam perguntas enviadas a eles e retornar resposta correlatas ao que foi proposto. Essa ferramenta foi utilizada para auxiliar estudantes de ciência da computação com Python e seus estudos relacionados à programação. Após isso, 98% das respostas foram precisas e 78% dos participantes relataram que o tutor IA auxiliou na sua aprendizagem. Os autores constataram que em uma turma que utilizou tutor inteligência artificial obteve aprovações maiores do que as turmas que não utilizaram o programa em 2 anos de comparação. Ambos os estudos têm semelhanças no uso de *LLM* para auxiliar programadores nas perguntas e dificuldades que surgem durante suas pesquisas. Desse artigo, será utilizado a fundamentação de que o uso de IA auxiliou os estudantes e alguns dos métodos utilizados para verificar a eficácia da LLM criada.

O estudo *Frankford et al.* (2024) mostra a viabilidade do uso de IA como tutor na aprendizagem de engenharia de software, utilizando o GPT-3.5 turbo para essa finalidade. Foi empregada a plataforma de código aberto *Artemis*, usada para avaliações automáticas de exercícios de programação. Utilizando a ferramenta Artemis, os autores conseguiram observar o comportamento dos estudantes ao submeterem códigos e identificarem quando recorriam a um modelo *LLM* para auxílio. Três perfis foram identificados: os que não utilizavam IA (Inteligência artificial), os que faziam perguntas antes de testar o código e os que questionavam após enviar o código para validação. Posteriormente, os autores aplicaram formulários e constataram que os resultados não apresentaram significância estatística. Para concluir o potencial relevante de *LLM* na educação em programação é inegável, mas ainda é necessário o refinamento de como realizar esse progresso junto da IA no ensino, sem perder o aspecto humano que é insubstituível na educação. Este estudo servirá como referência para evitar os erros cometidos no retorno dos *prompts* além de não terem feito mais avaliações e orientar estratégias que visem obter resultados positivos no aprimoramento da aprendizagem.

O estudo *Phung et al.* (2024) demonstra como obter retornos aceitáveis e menos genéricos a perguntas feitas por estudantes para o GPT-4. O estudo mostra que, ao utilizar duas versões do GPT (4 e 3), é possível realizar uma validação cruzada entre as LLMs. Nesse método, uma versão gera a resposta e a outra, geralmente a versão mais antiga,

atua como usuário para validá-la com base em parâmetros pré-definidos de aceitação. Caso o modelo que realiza o papel de usuário confirme que o retorno está adequado para o usuário, o texto é formatado para uma instrução didática refinada. Com esse processo os autores obtiveram melhora nos textos providos para o usuário e conseguiram evitar alucinações. Com isso obtiveram respostas concisas e precisas para as perguntas feitas ao usuário. O presente estudo se assemelha em construir *prompts* didáticos para as perguntas feitas pelos estudantes sobre uma nova *stack* a qual estão aprendendo. Será utilizado a metodologia de refinamento de *feedbacks* com algumas alterações para se adequar ao propósito do estudo atual. Contudo, dadas as conclusões obtidas pelos autores, esse artigo adotará uma versão do GPT que é factualmente menos "capaz" que outra para ser o usuário fictício.

O trabalho *Shrestha and Parnin* (2019) apresenta as três instrumentos (*Multiple Choice*, *Yes/No* e *Surprise*) para identificar equívocos cometidos por programadores ao migrarem entre linguagens, com base no conhecimento prévio de outras tecnologias. Foram utilizadas as linguagens Python e R e a pesquisa foi realizada com 32 graduados em Ciência da Computação. Como metodologia utilizando os 3 instrumentos para identificar equívocos entre linguagens, realizaram testes para obterem valores de acordo com as respostas dos usuários, validando se eram corretas. Primeiramente os autores fizeram ao todo 10 perguntas (5 de múltipla escolha e 5 de sim/não), após isso colocaram em um questionário essas perguntas intercalando entre múltipla escolha e sim/não. Como resultado, os autores descobriram que em questões de Sim/Não os acertos foram de 78% enquanto perguntas de múltipla escolha geraram resultados de 64%. O estudo conclui que programadores experientes podem aproveitar o conhecimento prévio, mas devem estar atentos a equívocos. Além disso, no estudo os autores citam que o uso de *dataframes* reduziu as barreiras na informação uma vez que os programadores podiam ver as suas predições instantaneamente. Este artigo será utilizado como base metodológica, especialmente na elaboração do questionário voltado a compreender se o uso de IA auxiliaria na aprendizagem de uma linguagens de programação.

No artigo *Li et al.* (2023) os autores analisaram como erros de *automated short-answer grader (ASAG)*, que são utilizados para correção de respostas, afetam a aprendizagem de alunos em questões de resposta curta sobre código. Os dados foram obtidos de 30 entrevistas e 40 questionários com estudantes. Erros de falsos positivos (FPs), quando respostas erradas são marcadas como corretas, foram os que mais prejudicam o aprendizado, pois os alunos não detectaram o erro nem revisaram o feedback. Já os falsos negativos (FNs), respostas corretas marcadas como erradas, só afetaram negativamente na aplicação por questionário, onde houve menor engajamento. Alunos com melhor desempenho foram mais afetados por FPs, por confiarem em suas respostas. Os autores concluem que é essencial projetar autores que estimulem a revisão crítica, mesmo após um acerto, e que se adaptem ao perfil do aluno. Será utilizado as análises em falsos positivos para embasar os conhecimentos acerca da necessidade de gerar *prompts* sem erros e como isso impacta na aprendizagem.

Por fim, os estudos mencionados servirão como base para a construção da metodologia e para validar hipóteses que ainda serão formuladas. No entanto, este trabalho não busca apenas replicar resultados anteriores, mas propor soluções para os obstáculos identificados, como as alucinações da Inteligência Artificial, e oferecer novas contribuições

sobre o impacto do uso da IA no processo de aprendizagem.

4. Materiais e Métodos

O presente estudo configura-se como um experimento controlado de natureza quantitativa e qualitativa, voltado para avaliar a utilização de LLMs para auxiliar na aprendizagem de uma nova linguagem. Essa abordagem permitirá considerar não apenas os dados e métricas, mas também as opiniões e percepções dos participantes sobre o processo. A escolha por esse tipo de pesquisa se justifica pela compreensão de que, apesar de os números serem relevantes, podem não traduzir fielmente a experiência vivenciada pelos candidatos [Frankford et al. 2024]. Assim, é essencial combinar dados quantitativos com relatos qualitativos, de forma a enriquecer a análise sob múltiplas perspectivas.

Um Sistema Tutor foi estruturado a partir de um conjunto de critérios definidos para orientar a atuação das duas inteligências artificiais (IAs): a IA principal, responsável por gerar as respostas ao estudante, e a IA validadora, encarregada de verificar se esses critérios foram seguidos. Sendo esses critérios: a IA não deve fornecer soluções completas nem trechos de código que resolvam o exercício; deve orientar o estudante explicando conceitos, lógica e sintaxe por meio de exemplos externos ao desafio;deve oferecer apenas orientações graduais, fornecendo exemplos progressivamente mais detalhados e dicas diretamente relacionadas ao contexto do exercício, acompanhando de forma incremental cada dificuldade identificada ao longo do processo, incentivando o raciocínio autônomo; e deve apontar e explicar erros básicos, como problemas de sintaxe ou indentação, sem apresentar a solução final. Além disso, deve adaptar o nível de explicação conforme o conhecimento do estudante realizando auto ajustes nos *feedbacks* até identificar que alcançou o grau de auxílio mais adequado ao seu progresso, garantindo clareza para iniciantes e objetividade para perfis mais avançados. Por fim, a IA pode sugerir melhorias ou boas práticas somente após o estudante concluir sua tentativa, sempre preservando sua autonomia no processo de aprendizagem.

4.1. Ferramentas

O ambiente de pesquisa foi composto por dois tipos de computadores. O primeiro, utilizado para a execução do Sistema Tutor, possuía 8 GB de RAM, processador Intel i5 de 8ª geração e 120 GB de armazenamento. Os demais computadores, pertencentes ao laboratório da PUC Minas, foram utilizados pelos estudantes para realizar os exercícios de programação, enviar consultas à IA por meio do WhatsApp e preencher os formulários. Para a validação cruzada das respostas geradas pelas IAs, foi empregado o modelo GPT-4o-mini. Além disso, fez-se necessário um código implementado em JavaScript para integrar as diferentes IAs, o uso do Google Forms para registrar as respostas relacionadas ao estudo e o acesso à plataforma LeetCode, que forneceu os desafios utilizados na execução das atividades.

4.2. Metodologia

Para o estudo, os participantes deveriam ter noções básicas de programação e estar aprendendo uma nova linguagem, o que correspondeu a alunos do 1º ao 3º período dos campi Coração Eucarístico e Lourdes do curso de Engenharia de Software. Programadores com senioridade inferior a pleno, foram priorizados, pois profissionais mais experientes tendem a carregar vieses de linguagens anteriores, o que pode interferir negativamente na

assimilação de novas sintaxes e paradigmas[Shrestha and Parnin 2019]. Porém não foi necessário fazer essa separação pois nenhum candidato se encontrava superior a junior. Com o objetivo de evitar vieses na análise e garantir uma amostra representativa, foram criados dois grupos de realização da atividade, sendo o primeiro o Grupo A que era composto por pessoas sem conhecimento prévio na linguagem e os candidatos do Grupo B que tinham algum conhecimento em Python.

A definição do tamanho da amostra foi realizada por meio do cálculo de amostragem evidenciado na Equação 1, considerando a proporção estimada (p) de 50%, tamanho populacional (N) 610 alunos, margem de erro (e) de 0,1 e valor crítico para o nível de confiança (Z) de 1,645, que corresponde a 90% de confiança. O valor obtido da fórmula corresponde a 61 participantes.

$$n = \frac{NZ^2p(1-p)}{(N-1)e^2 + Z^2p(1-p)} \tag{1}$$

Desta forma, a composição da amostra foi de 74 participantes voluntários que estavam aprendendo uma nova *stack* de programação, podendo ser programadores júnior, estagiários ou trainees, e o restante de pessoas que ainda não ingressaram no mercado de trabalho e se encontram em fase de estudo. Foi aplicada uma triagem nos dados por meio de formulário para selecionar apenas candidatos com o perfil adequado para a pesquisa.

Os *prompts* empregados para a geração das respostas pela LLM principal, bem como para a posterior validação dessas respostas pela LLM de validação, foram desenvolvidos com fundamento em boas práticas de engenharia de *prompting*, assegurando redução de ambiguidades na apresentação da informação, objetividade e adequação. Cada *prompt* foi cuidadosamente construído e alimentado com o contexto específico da atividade proposta, de modo a garantir uma melhor compreensão do problema por parte da LLM e, consequentemente, a geração de respostas relevantes, precisas e instrutivas.

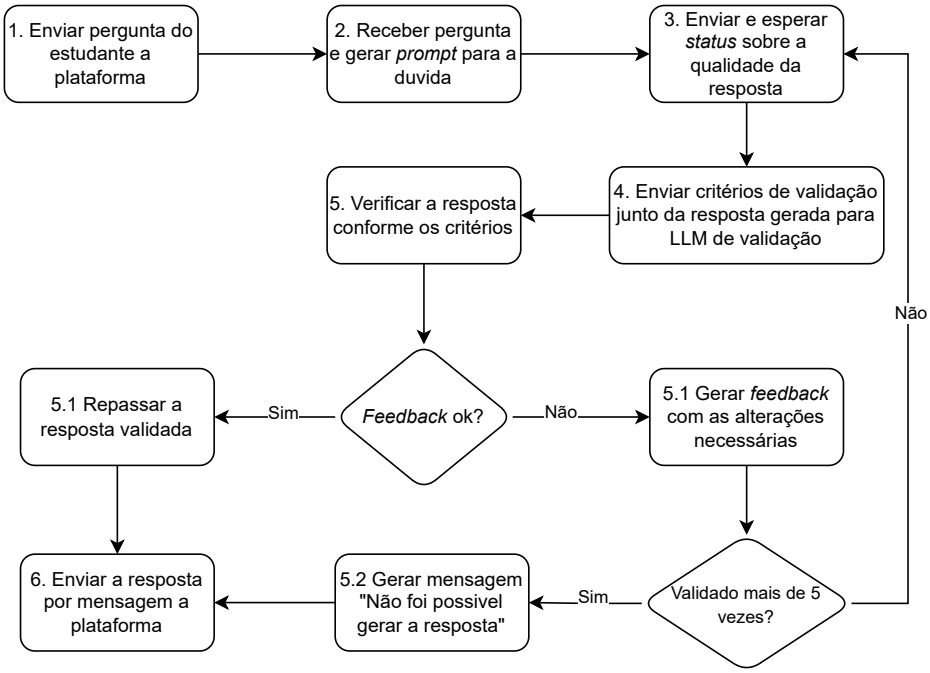

Figura 1. Fluxograma do ambiente em JavaScript que será criado.

A Figura 1 é representação visual de como foram feitas as etapas do Sistema Tutor e a utilização das LLMs dentro dele. Nos próximos parágrafos, detalha-se cada uma dessas etapas.

O Sistema Tutor gerencia o fluxo de comunicação entre o usuário e as LLMs. A pergunta enviada pelo estudante é encaminhada ao ambiente que faz integração com a LLM principal, responsável por gerar a resposta a partir do *prompt*. Em seguida, essa resposta é avaliada pela LLM de validação, que verifica se ela atende aos critérios estabelecidos. Se estiver adequada, é enviada ao usuário. Caso contrário, a LLM de validação solicita uma nova tentativa à LLM principal, repetindo o processo por até cinco ciclos. Se, após esse limite, a resposta ainda não estiver conforme, o sistema retorna uma mensagem padrão ao estudante, sendo ela “Não foi possível validar a resposta.”.

Foram aplicados três desafios de programação da plataforma LeetCode, um para cada nível de dificuldade: fácil, médio e difícil. O LeetCode foi escolhido por ser amplamente utilizado e por oferecer validação automática das soluções, utilizando três casos de teste públicos que verificam se o código está apto para ser avaliado pela plataforma. Esses testes permitem identificar se o participante efetivamente concluiu o desafio e, uma vez aprovados, possibilitam a obtenção de métricas como o tempo de execução, posteriormente empregadas na análise.

4.3. Métricas de Avaliação

Para avaliar o impacto do Sistema Tutor, adotou-se uma abordagem que combinou dados quantitativos de desempenho com a percepção qualitativa dos participantes. A principal métrica quantitativa utilizada foi a pontuação *Beats %* da plataforma LeetCode, que expressa o tempo de execução do código submetido em comparação ao de outros usuários

que resolveram o mesmo desafio na mesma linguagem, resultando em um *score* percentual que indica a proporção de códigos superados. Essa métrica é disponibilizada apenas quando a submissão passa por todos os casos de teste da plataforma, assegurando um nível mínimo de correção do código analisado. Complementarmente, aplicou-se um questionário baseado na escala *Likert* (de 1 a 5) para compor a dimensão qualitativa, no qual os participantes avaliaram a utilidade da IA, a coerência das respostas fornecidas, sua percepção da própria evolução no aprendizado e o grau de interesse em reutilizar o método de estudo.

Embora a métrica *Beats %* não represente um padrão-ouro para a análise de algoritmos, devido à sua natureza volátil e ao fato de refletir desempenho relativo em vez de complexidade assintótica (*Big* O), sua utilização mostra-se metodologicamente adequada ao escopo deste estudo. O objetivo não foi identificar a solução algorítmica ótima, mas investigar se o uso da IA poderia aproximar o desempenho de iniciantes ao de participantes com algum conhecimento prévio da linguagem, tornando-os comparáveis ou mesmo superiores. Nesse contexto, a métrica *Beats %* é pertinente porque só é disponibilizada após a aprovação completa do código pela plataforma, o que inclui a validação e a execução correta de todos os casos de teste públicos. Essa etapa de aprovação garante que apenas soluções funcionais e minimamente corretas sejam comparadas, permitindo avaliar, de forma consistente, o impacto do Sistema Tutor no desempenho final dos participantes.

4.4. Instrumentos Utilizados

Como instrumentos de coleta, empregou-se um questionário estruturado disponibilizado em plataforma digital, destinado a registrar a percepção dos participantes sobre o uso da IA e sobre o processo de aprendizagem. Para a interação com o Sistema Tutor, utilizou-se o WhatsApp como interface, permitindo que os estudantes enviassem suas dúvidas e recebessem orientações da IA em tempo real durante a execução dos desafios. Já os registros de desempenho quantitativo foram obtidos diretamente na plataforma LeetCode, responsável por fornecer as métricas de validação e tempo de execução dos códigos submetidos. O Sistema Tutor foi implementado em JavaScript, possibilitando a integração entre os modelos de linguagem e o ambiente de comunicação adotado, além de oferecer maior flexibilidade e capacidade de personalização da interação.

4.5. Realização do Experimento

O experimento foi realizado nos laboratórios da PUC Minas, os quais são compostos por computadores com configurações padronizadas. Cada candidato foi alocado em um desses computadores e orientado a utilizar apenas duas guias no navegador de sua preferência. A primeira guia permaneceu aberta com o aplicativo WhatsApp Web, destinado ao envio de dúvidas relacionadas aos desafios propostos. A segunda guia foi utilizada para acesso à plataforma LeetCode, responsável pela execução dos desafios disponibilizados no repositório GitHub deste artigo, cujo link encontra-se na Seção 8, bem como pelo envio dos códigos desenvolvidos e pela apresentação das correções fornecidas pela própria plataforma.

Além das orientações relacionadas ao uso das guias do navegador, os candidatos realizaram a leitura da mensagem inicial enviada pelo Sistema Tutor, a qual continha informações referentes à utilização adequada da ferramenta e ao fluxo adotado durante o experimento. Ao término da atividade, foi fornecido um código de desativação do Sistema

Tutor, responsável por disponibilizar o link do formulário de avaliação apresentado no Apêndice A, após isso o sistema foi desativado no chat de conversa do WhatsApp.

Por fim, os participantes foram orientados a ler atentamente todas as questões do questionário e a respondê-las cuidadosamente e com sinceridade.

4.6. Experimento Piloto

Para validar a metodologia e assegurar maior credibilidade ao experimento, foi conduzido um experimento piloto com a participação dos candidatos, sendo AB, MT e LY, avaliados de forma individual. Esses participantes recebem uma menção honrosa em reconhecimento ao empenho e à disponibilidade em contribuir voluntariamente com esta etapa do estudo, o que demonstra compromisso e colaboração essenciais para o aprimoramento da pesquisa.

A aplicação do piloto permitiu identificar ajustes necessários, como o aprimoramento do prompt da LLM para aumentar a assertividade das respostas, o tratamento de casos de exceção envolvendo usuários com menor familiaridade em Python, a otimização do retorno textual fornecido pela LLM e a implementação de controles para o uso da IA no teste final. Além de intervenções, foram realizados testes estruturais no código, que geraram a modularização das funções, a criação de arquivos de ambiente (Env) e a implementação de contexto para que a LLM mantivesse a "memória" da conversa.

Essa etapa foi essencial para o refinamento da ferramenta e contribuiu para garantir maior confiabilidade e eficiência no experimento.

5. Apresentação dos Resultados

Nesta seção, são analisados os resultados do experimento, iniciando pela apresentação dos dados quantitativos e qualitativos. A discussão integra a performance nos desafios com a percepção dos participantes para avaliar o impacto da IA no processo de aprendizagem.

É importante destacar que os grupos avaliados possuem tamanhos distintos, o que potencialmente influencia os resultados. Diante dessa assimetria, optou-se pela utilização dos testes U de Mann-Whitney e do Teste Exato de Fisher, uma vez que ambos são adequados para lidar com diferenças de tamanho amostral, evitando vieses nas análises estatísticas.

Nossa coleta de dados foi composta por um total de 74 respostas, sendo 57 do Grupo A e 17 do Grupo B. Em relação à maturidade técnica geral, os participantes foram distribuídos nas seguintes categorias: 55,4% foram classificados como "Estudante sem experiência profissional", 32,4% como "Estagiário/Trainee" e 12,2% como "Programador Júnior". Considerando especificamente a proficiência na linguagem utilizada no experimento (Python), 66,2% afirmaram não possuir experiência, 23% se declararam "Estudante sem experiência profissional", 5,4% se identificaram como "Estagiário/Trainee" e 5,4% como "Programador Júnior". Quanto à linguagem de programação estudada no momento da pesquisa, 5,4% indicaram Python, 1,4% Ruby, 40,5% Java, 4,1% C Sharp, 12,2% C++, 12,2% JavaScript e 24,3% escolheram a opção "Outra". Por fim, verificou-se que 97,3% dos estudantes (72 participantes) já haviam utilizado alguma ferramenta baseada em LLM para apoiar seus estudos, enquanto 2,7% (2 participantes) relataram não ter utilizado esse tipo de recurso anteriormente.

No que se refere à ocorrência de alucinações durante o uso do Sistema Tutor, observou-se que 91% dos participantes (68 estudantes) declararam não ter identificado nenhum tipo de alucinação ou não terem percebido esse fenômeno durante o experimento. Por outro lado, 8,1% dos participantes (6 estudantes) relataram ter percebido algum tipo de alucinação nas respostas fornecidas pela ferramenta.

Ao utilizar a métrica *Beats %*, foram estabelecidas faixas de valores para a análise dos resultados. Essas faixas foram definidas de forma arbitrária e correspondem a: Não realizou, Entre 0 e 50, Entre 51 e 70, Entre 71 e 80, Entre 81 e 90 e Acima de 90. As submissões classificadas como "Não realizou" foram excluídas da análise, uma vez que o próprio mecanismo do LeetCode condiciona a medição do tempo de execução à aprovação do código em, no mínimo, três casos de teste. Assim, a análise concentrou-se exclusivamente nos participantes que apresentaram uma submissão válida.

A Tabela 1 mostra os resultados de cada grupo por faixas.

Tabela 1. Comparativo de desempenho(Beats) — Grupo A e Grupo B

Faixa de pontuação	Grupo A			Grupo B		
	Fácil	**Médio**	**Difícil**	**Fácil**	**Médio**	**Difícil**
Não realizou	12	22	33	3	5	9
Entre 0 e 50	9	5	7	5	3	3
Entre 51 e 70	8	10	9	0	3	0
Entre 71 e 80	7	12	1	5	4	2
Entre 81 e 90	13	4	3	1	0	2
Acima de 90	8	4	4	3	2	1

A seguir, são apresentados os dados do formulário preenchido pelos candidatos dos Grupo A e B, refletindo suas percepções após a realização dos desafios.

Tabela 2. Percepção dos participantes do Grupo A em relação ao Sistema Tutor

Resposta	**Perguntas**					
	Explicações claras	**Útil para o aprendizado**	**Ajudou a corrigir erros**	**Interação natural**	**Melhorou o desempenho**	**Tempo suficiente**
Discordo totalmente	2	2	3	3	3	6
Discordo parcialmente	8	4	8	6	5	17
Concordo	20	19	14	21	18	19
Concordo parcialmente	15	17	17	13	9	6
Concordo totalmente	12	15	15	14	22	9

Tabela 3. Percepção dos participantes do Grupo B em relação ao Sistema Tutor

Resposta	**Perguntas**					
	Explicações claras	**Útil para o aprendizado**	**Ajudou a corrigir erros**	**Interação natural**	**Melhorou o desempenho**	**Tempo suficiente**
Discordo totalmente	2	2	2	2	2	2
Discordo parcialmente	2	2	3	3	3	3
Concordo	4	3	2	3	2	3
Concordo parcialmente	5	5	6	3	4	3
Concordo totalmente	4	5	4	6	6	6

Em relação à avaliação da evolução durante o uso do Sistema Tutor, o Grupo A apresentou 11 respostas classificadas como “Ruim”, 19 como “Neutro”, 20 como “Boa” e 7 como “Muito Boa”, não havendo registros para a categoria “Muito Ruim”; no Grupo B, os participantes distribuíram suas respostas em 1 para “Ruim”, 3 para “Neutro”, 10 para “Boa” e 3 para “Muito Boa”, também sem ocorrências em “Muito Ruim”. No que se refere à confiança para resolver mais desafios após a utilização do Sistema Tutor, o Grupo A registrou 38 respostas afirmativas e 19 negativas, enquanto o Grupo B apresentou 14 respostas afirmativas e 3 negativas. Por fim, quanto à disposição para utilizar novamente um método de estudo apoiado por uma LLM, 50 participantes do Grupo A responderam positivamente e 7 negativamente, ao passo que, no Grupo B, 16 afirmaram que utilizariam novamente e 1 declarou que não utilizaria.

5.1. Discussão dos resultados

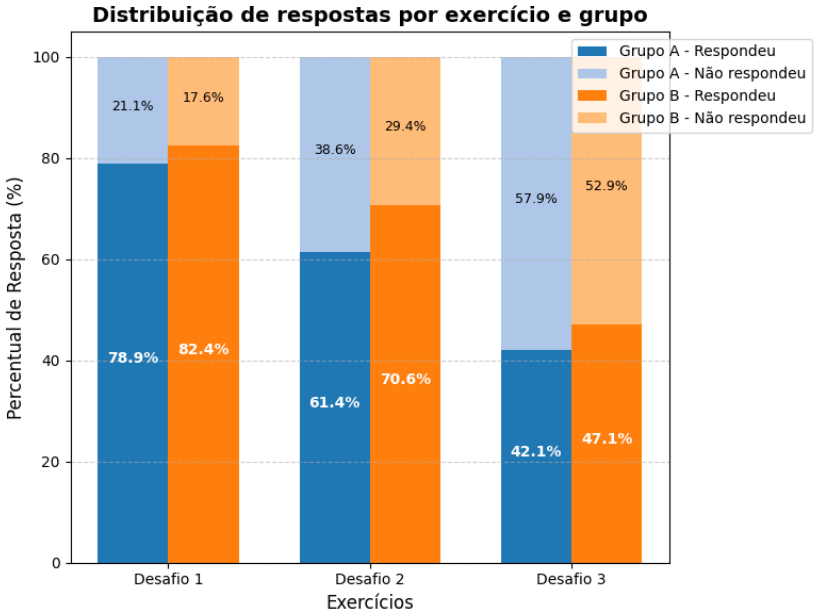

Figura 2. Gráfico de Barras Empilhado Comparativo de Resolução por Desafio

Na Figura 2 são apresentados os resultados obtidos por cada grupo em cada desafio, considerando a quantidade de candidatos que conseguiram ou não realizar a atividade proposta. No Desafio 2 observou-se uma taxa de resolução de 70,6% para o Grupo B e 61,4% para o Grupo A, resultado esperado devido ao uso do Sistema Tutor pelo Grupo B e ao maior conhecimento desse grupo na linguagem. O gráfico evidencia que a diferença de 9,2% no Desafio 2 é a maior entre os desafios, superando os valores de 3,5% e 5% observados no primeiro e no terceiro desafio, o que indica um melhor desempenho, conforme evidenciado pela Figura 2.

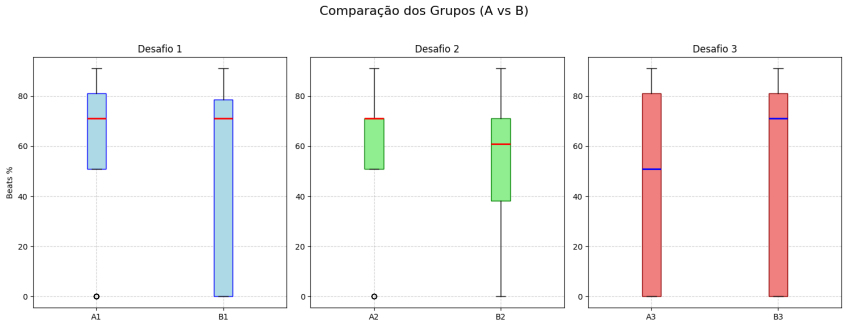

Figura 3. Gráfico de Boxplot Comparativo de Desempenho por Desafio

Após a realização do experimento, conduziu-se uma análise estatística por meio do teste U de Mann-Whitney, com o objetivo de verificar a hipótese de que o Grupo A apresentou desempenho superior ao Grupo B, com exceção do segundo desafio. Os resultados obtidos foram os seguintes: Desafio Fácil com p = 0,51, Desafio Médio com p = 0,58 e Desafio Difícil com p = 0,78. Como todos os p-valor estão substancialmente acima do limiar de significância de 0,05, não é possível rejeitar a hipótese nula. Assim, não foram encontradas evidências estatísticas de que o Grupo A tenha obtido desempenho superior ao Grupo B.

A visualização da Figura 3, que apresenta o boxplot de cada desafio, corrobora os resultados obtidos pelo teste de Mann-Whitney. No Desafio 1, observou-se que o Grupo A, composto por participantes iniciantes em Python com suporte do Sistema Tutor, e o Grupo B, que não utilizou essa ferramenta, obtiveram desempenho equivalente. Essa equivalência é demonstrada pela mesma mediana de pontuação (75) para ambos os grupos. No entanto, nota-se maior variabilidade nas notas do Grupo B, conforme ilustrado na Figura 3.

No Desafio 2, identificou-se um cenário distinto: o Grupo A obteve mediana superior (75) em comparação ao Grupo B (67,5). Considerando que o Grupo B possuía experiência na linguagem e utilizou o Sistema Tutor, era esperado que apresentasse melhor desempenho quanto ao tempo de execução do código. Além disso, observa-se que a dispersão das notas, representada pela altura do boxplot, do Grupo B foi menor em relação aos demais desafios, com desvio padrão de 33,84 no Desafio 2, frente a 39,91 no Desafio 1 e 41,37 no Desafio 3. Esse valor corresponde ao menor desvio padrão obtido pelo grupo.

No Desafio 3, classificado como o de maior complexidade, verificou-se elevada dispersão nos resultados de ambos os grupos, com desvio padrão de 34,51 para o Grupo A e 41,37 para o Grupo B. Nessa etapa, o Grupo B apresentou mediana superior à do Grupo A, com valores de 75 e 51, respectivamente. Esses resultados sugerem que, embora o uso do tutor de IA possa contribuir para o desempenho, tal recurso não substitui o desenvolvimento gradual do raciocínio lógico adquirido por meio de estudo contínuo e prática regular. Além disso, mesmo com o suporte oferecido ao Grupo A, a dispersão das notas não apresentou redução significativa, diferentemente do comportamento observado no Grupo B, conforme ilustrado na Figura 3.

5.2. Percepção dos Participantes

A análise das Tabelas 2 e 3 indica que ambos os grupos demonstraram percepções predominantemente positivas em relação ao Sistema Tutor. No Grupo A, composto por 57 participantes, as respostas concentraram-se nas opções favoráveis. Considerando o conjunto das seis perguntas, 71,6% das respostas estão distribuídas entre "Concordo" e "Concordo totalmente". As maiores proporções de avaliações positivas ocorreram em "Melhorou o desempenho" (70,1%) e "Interação natural" (61,4%). Em contraste, a pergunta "Tempo suficiente" apresentou o menor índice de concordância (56,1%) e concentrou a maior quantidade de discordâncias (40,3%), sugerindo que parte dos participantes percebeu limitação de tempo durante o uso da ferramenta.

No Grupo B, composto por 17 participantes, observa-se também um padrão favorável. Em média, 64,7% das respostas situaram-se entre "Concordo parcialmente" e "Concordo totalmente". As perguntas com maior proporção de avaliações positivas foram "Interação natural" (52,9%) e "Melhorou o desempenho" (58,8%). Embora os valores absolutos sejam menores devido ao tamanho reduzido do grupo, a distribuição revela percepção consistente de utilidade do Sistema Tutor.

A comparação entre os grupos mostra que o Grupo A apresentou variabilidade nas respostas, enquanto o Grupo B apresentou distribuição estável e concentrada em avaliações positivas. Esse comportamento pode estar relacionado ao nível de experiência prévia em Python entre os participantes do Grupo B, o que pode ter facilitado a compreensão das explicações fornecidas e reforçado a percepção de utilidade da ferramenta. De modo geral, os dados indicam que o Sistema Tutor foi bem avaliado nos dois grupos, com diferenças pontuais associadas sobretudo ao tempo disponível e ao nível de familiaridade prévia com a linguagem.

6. Ameaças à Validade

Nesta seção, apresentam-se as principais ameaças à validade identificadas no presente estudo, bem como as estratégias empregadas para mitigá-las durante a análise do impacto do Sistema Tutor baseado no processo de aprendizagem de programação.

Ameaças à validade externa: O tamanho desigual entre os grupos e a amostra composta exclusivamente por estudantes de Engenharia de Software limitam a generalização dos resultados. O estudo concentrou-se na linguagem Python, o que restringe a aplicação das conclusões a outros contextos. A possibilidade de uso de materiais externos pelos participantes também constitui uma ameaça, mitigada por orientações e monitoramento durante a realização dos desafios.

Ameaças à validade interna: O uso do WhatsApp como plataforma de interação pode ter introduzido distrações, interrupções e perda de foco, influenciando o desempenho dos participantes durante as atividades. Esse ambiente não controlado pode ter afetado a forma como cada participante recebeu e processou o feedback do Sistema Tutor.

Ameaças à validade de construção: A padronização dos prompts não impediu variações nas interações, uma vez que o conteúdo das dúvidas influenciou o detalhamento das respostas. O uso intercalado da IA entre os desafios resultou em diferentes níveis de exposição ao sistema, o que pode ter afetado a consistência das medidas analisadas.

Ameaças à validade de conclusão: A métrica *Beats %*, baseada apenas no tempo

de execução relativo das submissões, não avalia a qualidade das soluções nem o domínio conceitual, o que pode limitar a precisão das inferências sobre desempenho.

7. Conclusão

Este trabalho investiga o impacto do uso de uma LLM no processo de aprendizagem de uma nova *stack* de programação. Para isso, estabeleceram-se três objetivos principais: desenvolver um sistema de apoio ao estudo, aplicá-lo em um experimento controlado estruturado em desafios práticos e analisar sua eficácia por meio de dados quantitativos e qualitativos.

A análise quantitativa teve como finalidade testar a hipótese alternativa, segundo a qual o uso da LLM resultaria em desempenho superior em comparação ao grupo que não utilizou a ferramenta. Os resultados dos testes estatísticos, incluindo o teste U de Mann-Whitney, não forneceram evidências suficientes para rejeitar a hipótese nula, que assume a inexistência de diferença significativa entre os grupos. Dessa forma, os resultados numéricos não permitem afirmar que o uso da LLM tenha produzido melhora mensurável no desempenho técnico dos estudantes.

Os dados qualitativos, entretanto, mostraram que parte dos participantes percebeu benefícios subjetivos no processo de aprendizagem. Entre os relatos coletados, destacam-se clareza ao resolver os desafios, sensação de evolução ao longo da atividade e intenção de utilizar novamente ferramentas baseadas em LLM. Esses elementos sugerem que, embora não tenham sido identificados ganhos quantitativos, a ferramenta pode influenciar aspectos relacionados à experiência de estudo, e pode influenciar subjetivamente o processo de aprendizagem, mesmo sem apresentar impacto imediato e mensurável no desempenho prático.

Os resultados obtidos reforçam a necessidade de ampliar a discussão sobre o papel das LLMs no ensino de programação. A integração dessas ferramentas ao ambiente educacional pode funcionar como apoio complementar ao docente, ao mesmo tempo em que demanda novas estratégias de orientação, acompanhamento e supervisão.

Considerando as limitações identificadas, recomenda-se que trabalhos futuros ampliem e refinem as métricas de avaliação, incorporando indicadores relacionados à qualidade das soluções produzidas. Sugere-se também o aprimoramento da interface de interação do Sistema Tutor, a fim de reduzir distrações e aumentar o controle sobre o envio de dúvidas. Por fim, a ampliação do tamanho da amostra pode fortalecer a validade estatística e permitir análises mais robustas sobre os efeitos das LLMs na aprendizagem de programação.

8. Pacote de Replicação

O pacote de replicação encontra-se disponível em: https://github.com/ICEI-PUC-Minas-PPLES-TI/plf-es-2025-1-tcci-0393100-pes-arthur-jansen/tree/main

Referências

Bang, Y., Cahyawijaya, S., Lee, N., Dai, W., Su, D., Wilie, B., Lovenia, H., Ji, Z., Yu, T., Chung, W., Do, Q. V., Xu, Y., and Fung, P. (2023). A multitask, multilingual, multimodal evaluation of ChatGPT on reasoning, hallucination, and interactivity. In

Park, J. C., Arase, Y., Hu, B., Lu, W., Wijaya, D., Purwarianti, A., and Krisnadhi, A. A., editors, *Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 675–718, Nusa Dua, Bali. Association for Computational Linguistics.

Brasil, C. (2025). Ia alcança 93% dos brasileiros, mas apenas 54% entendem o que é, diz estudo. Acesso em: 03 set. 2025.

Frankford, E., Sauerwein, C., Bassner, P., Krusche, S., and Breu, R. (2024). Ai-tutoring in software engineering education. In *Proceedings of the 46th International Conference on Software Engineering: Software Engineering Education and Training*, ICSE-SEET '24, page 309–319, New York, NY, USA. Association for Computing Machinery.

Jiang, X., Dong, Y., Wang, L., Fang, Z., Shang, Q., Li, G., Jin, Z., and Jiao, W. (2024). Self-planning code generation with large language models. *ACM Trans. Softw. Eng. Methodol.*, 33(7).

Li, T. W., Hsu, S., Fowler, M., Zhang, Z., Zilles, C., and Karahalios, K. (2023). Am i wrong, or is the autograder wrong? effects of ai grading mistakes on learning. In *Proceedings of the 2023 ACM Conference on International Computing Education Research - Volume 1*, ICER '23, page 159–176, New York, NY, USA. Association for Computing Machinery.

Ma, I., Krone-Martins, A., and Videira Lopes, C. (2024). Integrating ai tutors in a programming course. In *Proceedings of the 2024 on ACM Virtual Global Computing Education Conference V. 1*, SIGCSE Virtual 2024, page 130–136, New York, NY, USA. Association for Computing Machinery.

Phung, T., Pădurean, V.-A., Singh, A., Brooks, C., Cambronero, J., Gulwani, S., Singla, A., and Soares, G. (2024). Automating human tutor-style programming feedback: Leveraging gpt-4 tutor model for hint generation and gpt-3.5 student model for hint validation. In *Proceedings of the 14th Learning Analytics and Knowledge Conference*, LAK '24, page 12–23, New York, NY, USA. Association for Computing Machinery.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI Technical Report*. Accessed: 2025-06-24.

Shrestha, N. and Parnin, C. (2019). Instrument designs for validating cross-language behavioral differences. In *2019 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, pages 205–209.

Smolić, E., Pavelić, M., Boras, B., Mekterović, I., and Jagušt, T. (2024). Llm generative ai and students' exam code evaluation: Qualitative and quantitative analysis. In *2024 47th MIPRO ICT and Electronics Convention (MIPRO)*, pages 1261–1266.

Wang, J. and Chen, Y. (2023). A review on code generation with llms: Application and evaluation. In *2023 IEEE International Conference on Medical Artificial Intelligence (MedAI)*, pages 284–289.

White, J., Fu, Q., Hays, S., Sandborn, M., Olea, C., Gilbert, H., Elnashar, A., Spencer-Smith, J., and Schmidt, D. C. (2023). A prompt pattern catalog to enhance prompt

engineering with chatgpt. In *Proceedings of the 30th Conference on Pattern Languages of Programs*, PLoP '23, USA. The Hillside Group.

Apêndice

A. Formulário Pós Desafio

Avaliação do Sistema Tutor com IA durante a Realização de Desafios de Programação

1. Dados do Participante (Triagem Inicial)

1.1. Nome (opcional): _____

1.2. Você já possui experiência prévia com programação?
( ) Sim ( ) Não

1.3. Qual linguagem de programação você está atualmente estudando? _____

1.4. Em qual dos perfis abaixo você se encaixa?
( ) Estudante sem experiência profissional
( ) Estagiário / Trainee
( ) Programador Júnior
( ) Outro: _____

1.5. Você já utilizou ferramentas de IA anteriormente para auxiliar nos estudos?
( ) Sim ( ) Não

2. Experiência com o Sistema Tutor

Para as questões abaixo, utilize a escala de 1 (Discordo totalmente) a 5 (Concordo totalmente):

2.1. A IA forneceu explicações claras e compreensíveis.
( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5

2.2. As respostas da IA foram úteis para meu aprendizado.
( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5

2.3. A IA ajudou a identificar e corrigir meus erros.
( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5

2.4. Em algum momento, a IA apresentou respostas incoerentes ou com informações incorretas (alucinações)?

( ) Sim ( ) Não

Se sim, descreva brevemente o ocorrido:

2.5. A interação com a IA foi natural e fácil de usar.
( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5

2.6. Considero que a IA melhorou meu desempenho nos exercícios.
( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5

2.7. O tempo total do experimento (máximo 1 hora) foi suficiente.
( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5

3. Avaliação Final

3.1. Como você avalia sua própria evolução após os exercícios? Utilize a escala de 1 (Muito ruim) a 5 (Muito boa):
( ) 1 ( ) 2 ( ) 3 ( ) 4 ( ) 5

3.2. Você se sente mais confiante para resolver desafios de programação após a experiência com o sistema tutor?
( ) Sim ( ) Não ( ) Em parte